

TITLE OF THE INVENTION

AUTOMATED SYSTEM FOR ANALYZING CHARGES AND CREDITS OF BANKS
AND OTHER FINANCIAL INSTITUTIONS

COPYRIGHT NOTICE AND AUTHORIZATION

5 Portions of the documentation in this patent document contain material that is
subject to copyright protection. The copyright owner has no objection to the facsimile
reproduction by anyone of the patent document or the patent disclosure as it appears in
the Patent and Trademark Office file or records, but otherwise reserves all copyright
rights whatsoever.

10 BACKGROUND OF THE INVENTION

 Over the last few decades, banks and other financial institutions, hereafter
collectively referred to as "banks," have begun charging their clients for the services
performed. These charges include, but are not limited to: deposit ticket charges, night bag
deposits, electronic transmittal of information, returned check charges and postage.
15 Banks no longer rely to the same extent as they have in the past on the income generated
from loans. Bank officers have even attended seminars strictly targeted to increasing
revenue through service charges. Even if the evolution of service charge revenue was
unintentional, it has become a lucrative opportunity for many banks.

 Clients often receive cryptic monthly statements as to what charges they are
20 actually paying for. Consider a scenario wherein supermarket aisles did not indicate the
price of the food on the shelves. In addition, suppose the receipt did not include a detail
of the food that was purchased or the price of each item. Instead, only a total due during
that shopping trip was provided. Or, if the detail was included, it was not listed in readily
understood terms, but only in the UPC Barcodes of the product. It would not be easy to
25 comparison shop, or to determine better ways to shop, such as buying larger sizes of
items you may use frequently. This scenario is analogous to the way service charges are
presently handled between banks and their clients. Furthermore, the bank often deducts

service charges from the earnings credit (Interest) posted to the clients' accounts. To continue the supermarket analogy, imagine if the receipt didn't tell you how much you actually spent that day, just a running total against some prepaid balance you may have on account with the supermarket. Banks may not have intentionally created these problems. However, these problems prevent clients from understanding and managing their charges effectively.

These analogies illustrate how many large businesses (clients) currently interact with their banks. Coupled with the fact that charges imposed are not listed in layman's terms on the bank analysis, it has become extremely difficult for a client to control these charges. This weakens the client's financial and bargaining position and gives little leverage when negotiating with the client's bank. It also inhibits the client from attaining the highest possible earnings credit.

Many clients have numerous locations (stores) throughout the country who use the same bank. The statements issued by the bank offer little help in determining charges per store, and do not provide the client's financial managers with the essential data required to make appropriate fiscal decisions.

An additional problem for clients is that each bank uses its own set of terms to define its service charges and credits, thereby making it difficult or impossible for a client to comparison shop among banks for the best deal on service charges and credits.

Accordingly, there is an unmet need for analysis tools to allow a client to more fully understand bank service charges and credits. The present invention fulfills such a need.

SUMMARY OF THE INVENTION

A first embodiment of the present invention provides a computer-implemented scheme for preparing bank service charge reports for banking activity of a client. In the scheme, an electronic translator is provided that converts service charge items of a plurality of individual banks, expressed in terminology of the respective individual banks, to a service description expressed in a standardized terminology. At least some of the service charge items of different banks have the same standardized service description. Bank service charge items and bank account data are inputted into a computer. The bank service charge items and bank account data are for a specified period of time for one or

more banks being used by a client. The bank service charge items and bank account data are obtained from one or more bank statements of a client and are expressed in the one or more statements in terminology used by the one or more banks. In the computer, one or more bank service charge reports are automatically created for the client from the
5 inputted bank service charge items by using the electronic translator. Each bank service charge report provides a breakdown of bank service charges based upon the standardized service descriptions.

A subset of the standardized service descriptions of bank service charges may define a total amount of a predefined type of bank service charge costs. In this scheme,
10 the total amount of a predefined type of bank service charge costs is calculated by adding together the bank service charges in the subset. A total cost report is then created for the predefined type of bank service charge costs from the calculated total. The subset may define total depository costs, total check costs, total account maintenance costs, or any total of a service charge category.

A historical database may be provided of average bank service charges for
15 selected standardized service descriptions based upon bank service charges of a plurality of clients. The individually broken down bank service charges may then be compared to the average bank service charges. The comparison may be used by the client to identify potentially excessive bank service charges. Alternatively, a database of bank service
20 charges may be provided for each of the standardized service descriptions based upon bank service charges of all of the banks used by the client. The individually broken down bank service charges for one of the client's banks may then be compared to the bank service charges for other banks used by the client. Again, the comparison may be used by the client to identify potentially excessive bank service charges.

One of the standardized service charge items may be deposit ticket costs, wherein
25 deposits have a predefined earnings credit rate. The deposit ticket costs and the earnings credit rate may be used to calculate a breakeven deposit amount wherein the earnings credit exceeds the cost of depositing money.

In a second embodiment of the present invention, a similar process as described
30 above is performed for earnings credit items.

BRIEF DESCRIPTION OF THE DRAWINGS

The file of this patent contains at least one drawing executed in color. Copies of this patent with color drawings will be provided by the Patent and Trademark Office upon request and payment of the necessary fee.

5 Fig. 1 (provided in parts 1 and 2) summarizes reports generated by the present invention, and the information provided by the reports that is not currently available to a banking customer;

 Figs. 2-14 show examples of the reports summarized in Fig. 1;

10 Fig. 15 is a schematic block diagram of a cross-reference table in accordance with the present invention;

 Fig. 16 shows an entry screen of a form used to collect and store service charges;

 Figs. 17-19 show the contents of tables used in the cross-reference table;

 Fig. 20A and Fig. 20B show screen displays for viewing bank description and service description data in the cross-reference table;

15 Fig. 21 shows a table that stores client data;

 Fig. 22 is a flowchart of a history compilation process of the present invention;

 Fig. 23 shows the contents of a Bank Service Charge History table used for the history compilation process;

20 Figs. 24 and 25 shows the contents of tables used in a dynamic compilation process;

 Figs. 26 and 27 are data entry screens used in the report generation process;

 Fig. 28 shows the contents of a Category table which is used to generate the report in Fig. 7;

25 Fig. 29 is a screen display for selecting categories that the user wishes to appear in a report;

 Fig. 30 is a screen display for selectively choosing any combination of clients, banks, and/or service charges that the user wishes to appear in a report;

Figs. 31 and 32 show excerpts from charts that are used to determine costs of a particular service charge category; and

Fig. 33 is a screen display of a Working Sheet Form that is used to summarize service description items, and their actual costs and potential cost savings.

5 DETAILED DESCRIPTION OF THE INVENTION

Certain terminology is used herein for convenience only and is not to be taken as a limitation on the present invention. In the drawings, the same reference letters are employed for designating the same elements throughout the several figures.

DEFINITIONS

10 Comp Balance - balances on hand at bank, subject to the bank's earnings credit.

Earnings Credit – interest earned on available client's assets, taken after the Federal Reserve requirement is deducted.

Federal Reserve Requirement – amount by law which must be reserved by the bank for the Federal Reserve (non-earning asset), usually 10% of client's balance.

15 Treasury Bill – (T-Bill) – interest rate set by the US Government.

Fed Funds Rate - A target interest rate for banks borrowing reserves among themselves. It is set by the Federal Open Markets Committee. See web site:
<http://www.bankrate.com/brm/green/define/#fomc> for more information.

Float – checks which have been deposited but funds are not yet available.

20 Revision Date – date of the bank statement (usually issued on monthly basis).

Basis Point - one one-hundreth of a percent, e.g., 300 basis points is 3%.

OVERVIEW OF PRESENT INVENTION

The present invention is implemented as a software program called "The Bank Analyzer," that aims to empower clients with the necessary information to ensure fair
25 treatment from their bank. The software generates numerous reports which explain in layman's terms exactly how bank service charges are broken down. Furthermore, the reports are presented in color, making it easy for even a novice to elicit the needed

information. Armed with such information, a client will understand exactly where and how service charges can be reduced and/or earnings credits can be increased.

Prior to the present invention, some clients may have suspected that certain bank charges were askew, certain transactions were wasteful, and certain transactions were unnecessary. However, there was never any solid clear-cut method of substantiating these suspicions. Furthermore, it was the belief of clients that the resources necessary to even begin an investigation would outweigh any results. The present invention includes a plurality of analysis techniques that were previously not available and which are implemented in formulas that generate Illumination Reports (described below). The Illumination Reports yielded by the present invention provide a catharsis to the clients in managing their bank charges.

The basic philosophy of the invention is (1) Effective Management of Bank Service Charges; (2) Illumination of Data; and (3) Edification of the Client.

In the examples provided in the figures and tables below, Bank Names and Client names are fictitious, but the values and dollar amounts represent actual data.

Illumination Reports Table (Diagnostic Section)

Fig. 1 (provided in parts 1 and 2) summarizes in table form each report from the diagnostic section of the present invention, highlighting the areas of most impact. In many cases, clients are unaware that the data produced by the present invention may be significant to the financial well being of their company. In other cases, there is no simple method of obtaining this data. Lack of data precludes effective management of such charges.

The disclosure below covers various methods and systems to assist clients in managing their bank service charges and reducing or eliminating cost incurring transactions. The focus also centers on elevating client awareness and providing a clear-cut analysis of how bank charges are affecting their bottom line. Also, the disclosure presents the algorithms of the software which provide this assistance. Exemplary methods include:

1. Revealing actual Bank Service Charges assessed.
2. Minimizing strap charges, i.e., the charges a bank assesses for counting clients money, by strapping in denominations which yield the best results.

3. Comparison of clients' bank charges, i.e., Deposit Tickets, Night Bag Deposits, etc... with charges of other clients' banks nationwide. Recommendations are made based on these national averages.

4. Evaluation of clients' non-earning assets and recommendations on how to maximize revenue from these assets.

Exemplary algorithms include:

1. Using dynamic report generation forms to allow the user to extract the necessary information to execute the methods.

2. Creating cross-reference tables between a bank charge and its description according to the particular bank, and the actual charge expressed in layman's terms.

3. Storing all bank analysis data in a minimum number of tables, and calculating and deriving comparison information "on the fly," that is, each time a report is generated. This ensures up-to-the-minute accurate information.

DETAILED DISCLOSURE

This section outlines what each report accomplishes. Refer to the attached figures for examples that correspond to the figure numbers listed. The alphanumeric characters refer to data highlighted in the respective figure. For example, "2a" highlights data in Fig. 2.

Bank Balance Data (Fig. 2) – Here the Bank Analyzer exposes the total service charge assessed a client for the particular revision date. (2a) Note that the Bank Analyzer reveals the actual charge, not the 'net' one which the client now sees. The net charge includes a 'comp' balance, which are client's balances subject to the Banks' earnings credit. In some cases, the earnings credit offsets any service charges. That is, the client believes that no service charges are being paid, when, in fact, the service charge is deducted from any earnings credit due. Fig. 2 shows how this client (SUPER-CON Convenient Stores) has paid \$57,192.50 in the month of October, 1999 to their banks. The report details not only totals but amounts to each bank, e.g., \$15,035.17 paid to the Western Savings Co. (see line #2 of the example report)

Earnings Credit Analysis (Fig. 3)

This report color codes basis point shaved, to indicate abnormal bank activity.

This report is a scorecard to see how a client's bank performs against the T-bill rate. (3a)
It indicates the difference between T-Bill and client's actual earnings. The results are
instantly revealed. This process would consume hundreds of man-hours to produce
otherwise. Armed with this report, a client can be sure to be credited with at least the
5 T-Bill Rate.

Fed Funds (3b) – This same report also shows the amount of money the bank is
earning by investing the client's money at the Fed Funds rate. The bank will often invest
in Fed Funds when they are not using it to generate loan revenue. This gives the client a
look at how much money the bank can earn from the client's money.

10 Income Difference (3c) – Finally, the income difference is determined, which is
the difference a client can add to its bottom line if given the corresponding interest rates.
This generally results in a tremendous increase to the client's bottom line. (3d)

The Bank Names, as well as the basis points shaved, are printed in degrees of
color. This facilitates isolating the cases where the client can realize the maximum
15 benefit. (3e)

Earnings Credit And Negative Collected Rates (Fig. 4)

As a supplement to the Earnings credit Report, this report matches the Earnings
Credit on positive balances to the rate imposed on Negative Collected funds. Clients
should be charged the prime interest rate. However, typical findings indicate that clients
20 are sometimes charged as much as 300 basis points over prime. A fair bank would
actually pay at a 1:1 ratio by posting earnings at the same rate imposed for the negative
collected balance. The example report shows one sample bank (4a) which is charging
significantly higher negative collected rates than earnings credit rates, and one sample
bank (4b) which is charging fair rates.

25 Checks, Deposits And Float Data (Fig. 5)

Avg Check Size Per Store (5a)

Occasionally, the bank will assess a float table that does not correspond to actual
availability. This report will alert client to this fact, by showing them the Avg Check Size
per store. The client usually has a good idea of the average check a store receives, (e.g., a
30 drug store may receive checks that average about \$75, whereas a lumberyard may have

checks averaging \$300). If this check size becomes inflated, that is, if the client sees that the value is more than it normally would be, then the client knows to investigate and to look at each check. In many cases, the client may realize that the bank is not posting the checks to his/her account promptly. Without the use of the present invention, discovery of this fact would consume inordinate amounts of time and resources.

Out Of District (5b)

The Out Of District column indicates checks received from customers of the client that are drawn on banks outside the local district of the depository bank. It is common that most customers bank locally. Therefore, 90% should be in district. The column indicating 2 Day % is the derived Out Of District number. If this number is not within this normal average, it is an alert that the bank may not be providing the proper calculation with the float. Also, it is likely that the service charge is increased, because Out Of District service charges are generally more expensive.

Checks On Us (5c)

This column determines the presence that a bank has in a region, and On Us Checks should be treated as cash and should be made available immediately. Clients can now ensure that this is the case.

Breakdown By Service Charge Groups (Fig. 6)

The Bank Balance Data report illuminated the actual Total Service Charges assessed. This report begins putting the Total Service Charge number under the microscope, commonly referred to as drill-down reporting in the computer field. This report shows the 3 main categories:

Categories Depository Costs (dep tickets, cash bags, etc...) (6a)

Check Costs (Checks dep, ret items, etc..) (6b)

Account Maintenance (balance reporting) (6c)

The percentage of the total cost is analyzed, and if these three categories do not total close to 100%, then there is a good chance that a particular charge is askew.

Potential items to investigate include:

1. Negative interest charge for using uncollected funds (See Earnings Credit and Negative Collected Rates report)

2. Bounced check fees - too high?

These percentages can be compared with other clients' percentages to assist in evaluation. Volumes (amount of activity) are also considered, to ensure fair comparisons.

Breakdown By User Selected Groups (Fig. 7)

This report allows for the creation of custom reports tailored to each client.

5 Multiple service charges can be lumped into groups, and then these groups can be evaluated. For example, a client may want to see a breakdown of lock box deposit charges. Presently, this would require mining every statement for any charge pertaining to a lock box. It is not always clearly indicated on the bank statement, so research could also be required. Using a conventional analysis process, it could require numerous hours
10 or more for a client to presently determine all of the lock box charges. In the present invention, service charges can be grouped into a category, for example, Lock Box Charges. Then, the cross-reference table links all lock box charges: Correspondence, Courier, Delivery US Mail, Checks Deposited, etc... The report then displays at a glance the actual Lock Box charges that a client is paying for.

15 Lock Box Charges is just an example category. Any group can be analyzed, e.g., Depository Costs, Checks, Cash, Coin, etc... This information is invaluable to the client and has never before been available to the client. Any categories can be supplied, yielding hundreds of possibilities for analysis depending on the client's particular situation.

20 Unit Price Summary (Fig. 8)

This report shows every service charge assessed for the client encompassing all stores, all banks, and every line item. The report shows at a glance monthly totals, yearly totals (calculated) and also average and weighted average, which considers volume. This information was so difficult to obtain using conventional analysis processes that clients
25 never even considered attempting to acquire it.

Deposit Ticket Costs (Fig. 9, provided in parts 1 and 2)

This report examines the prudence of daily deposits by examining every cost associated with making a deposit. Since clients make frequent deposits of large sums of checks and cash, these charges can be substantial. See 9a for a sample of the total cost
30 incurred in one year for one client. Prior to the present invention, these charges could

never be investigated or verified, and therefore it was not possible to show inefficiencies in the client's business practices. This report also determines the daily deposit a client must make in order to "break even." It is actually possible for a client to lose money by making too many deposits, because the cost of depositing the money outweighs any earnings credit that may be earned. (9b)

Finally, the Potential Daily Loss columns show the exact amount lost (9d) if the deposit were equal to the column headings. (9c) It should be noted that these values depend upon the average return rate a client expects to make. (9e) This value is dynamic, and can be changed (inputted) each time the report is generated.

Cash Activity Costs (Fig. 10)

Banks charge their clients to count the cash being deposited. Clients are generally charged in one of two ways: by strap or per \$1000 counted. If a particular client receives a lot of cash, it behooves a client to be charged by strap since 1000 single dollar bills is charged the same as 10 one hundred dollar bills, even though it is more labor intensive to count 1000 bills than 10 bills. When charged by strap, 100 100 dollar bills are charged the same as 100 1 dollar bills. This decision varies depending on the cash composition of the business client is engaged. For example, a newsstand which receives numerous one dollar bills is generally better off paying by the 1000, because they will have numerous bills. On the other hand, a supermarket may be better off paying by strap, as they may receive greater denominations. In either case, the reports will highlight this information, permitting the client to choose wisely.

Another area often overlooked by clients is the Rolled coin charges (10a). Clients are often charged for the phone call to order the coin! This report allows deeper probing into this matter.

Another benefit of the Cash Activity Costs report is that it allows for verification that the proper amount of cash is being counted. Currently, clients must assume that the banks' report is correct.

To gain insight into just how much it costs to count cash, see the client's monthly charge (10b) and yearly projection. (10c)

Banking Activity (Fig. 11)

This is a summary report, which details vital information such as Deposit Tickets, BAI Detail, Night Bags and Rolled Coin charges. It is used as a support and guide to aid the client in understanding the detail diagnostic reports.

5 Analysis By Unit Price (Fig. 12)

This report takes the Unit Price Summary and drills down every charge to list each bank where the charge was derived. It is a lengthy report and is used to isolate problem areas when the summary indicates a potential problem area. It serves as a complete audit trail tracking the lowest level of detail from each of the client's banks.

10 Illumination Reports (Action Items) – (Figs. 12-14)

This section provides a 'blueprint' to the client for the steps which should be taken.

This section also contains every line item of every analysis from every bank for the client, and the revision date being considered. (see Fig. 12) This serves as supporting information when the client is making the decisions necessary to eliminate or reduce transactions, as well as negotiate with the bank for a reduction in charges which are deemed unfair.

This section contains the following items:

20 Working Sheet Report (Fig. 13) – contains a trace number which references where the original data came from. Provides client with actual volume and cost amounts, in addition to the recommendations based on the comparison with similar clients at similar banks. It also highlights the potential savings a client can garner from implementing these recommendations. Annual projections are extrapolated.

25 Projected Savings Report (Fig. 14) shows the clients' potential savings for each of their banks if the recommendations in Fig. 13 are implemented.

How the Bank Analyzer Works

Examples of source code for implementing the Bank analyzer are provided in this section.

There are 4 sections to the Bank Analyzer, as follows:

SECTION I - Static Compilation

SECTION II - Dynamic Compilation - Input of Monthly Bank Analysis Data

SECTION III - Report Generation - Black Box effect

5 SECTION IV - Illumination - Meeting with Client

This portion of the disclosure provides a detailed description of how the present invention works, and how it generates the reports to edify the clients.

SECTION I – Static Compilation

Static Compilation refers to the acquiring of data and inputting of data which will
10 form the basis of the eventual evaluation prepared for the client. This section has two functions. The first function is to build a master cross-reference table, shown schematically in Fig. 15. The second function is history compilation.

Function #1 - Building A Master Cross-Reference Table

This function begins with the massive input of every possible service charge from
15 every bank a client is associated. Each service charge is assigned a unique code and a layman's (layperson's) term. Fig. 16 shows an entry screen of a form used to collect and store service charges. Every possible charge incurred by the clients is entered here, and is assigned a standardized service description.

Fig. 17 shows a layout and structure of the cross-reference table. This table
20 houses every bank service charge encountered from the statements. The ServiceCodes are a unique way of identifying all charges for all clients. This table can add new entries as banks create new service charges, but the information is generally static and therefore this table is generated during the Static Compilation section of the Bank Analyzer software.

Once the service codes are established, it is necessary to maintain a database of
25 bank information and a link between the bank's service charges and the internal codes of the Bank Analyzer. This ensures that clients will see the same terminology when analyzing all of their banks, and will also provide the method of translating back to the bank's language when dealing with a specific bank. These functions are accomplished mainly by the use of two tables, Banks (Fig. 18) and BankServ (Fig. 19). Figs. 17 and 18
30 combine to make the cross-reference table. The Banks table houses in a computer

database each bank that a client is associated. The BankServ table houses in a computer database each service charge assessed by a bank, and the corresponding code assigned to this service charge by the Bank Analyzer software. The BankDescription field maintains the terminology used by the bank for future reference when dealing with the bank. This foundation data is collected in a computerized form and can be searched and retrieved via screen displays as shown in Fig. 20A and Fig. 20B.

Fig. 20A and Fig. 20B illustrate the use of an electronic translator that converting service charge items of a plurality of individual banks, expressed in terminology of the respective individual banks, to a service description expressed in a standardized terminology. The translation goes in the reverse direction when the results are used in discussions with the clients' banks. At least some of the service charge items of different banks have the same standardized service description. For example, the service charge for "Deposit Tickets" (standardized service code 00425) is referred to as "Deposited Processed" by Eastern Savings Co. (see Fig. 20A), and is referred to as "Credit Posted" by Commercial Bank (see Fig. 20B). In another example, "BAI Detail" (Bank Administration Institute Detail) (standardized service code 00973) is referred to as "Dep Recon Proc" by Eastern Savings Co. (see Fig. 20A), and is referred to as "Recon" by Commercial Bank (see Fig. 20B).

Fig. 21 shows a Clients table that stores client data for the Bank Analyzer software.

Sample Visual Basic for Applications (VBA) code for controlling interface of Banks table:

Code

```
1  VERSION 1.0 CLASS
2  BEGIN
3    MultiUse = -1 'True
4  END
5  Attribute VB_Name = "Form_Banks"
6  Attribute VB_GlobalNameSpace = False
7  Attribute VB_Creatable = True
8  Attribute VB_PredeclaredId = True
9  Attribute VB_Exposed = False
10 Option Compare Database
11 Option Explicit
12
13 Private Sub Command16_Click()
14   D Cmd.OpenRep rt "Banks By Code", acViewPreview
15 End Sub
16
17 Private Sub Command17_Click()
```

```

18 D Cmd.OpenRep rt "Banks By Name", acViewPreview
19 End Sub
20
5 21 Private Sub Command52_Click()
22 D Cmd.OpenReport "Banks Quick Print", acPreview, , "[BankC de] = '" & Me!BankC de &
23 End Sub
24
10 25 Private Sub Form_Activate()
26 DoCmd.Restore
27 End Sub
28
15 29 Private Sub Form_Current()
30 If Me.NewRecord Then
31 BankCode.SetFocus
32 End If
33 End Sub
34
20 35 Private Sub Form_Load()
36 If Not IsNull(Me.OpenArgs) Then
37 Me!BankCode = Me.OpenArgs
38 End If
39 End Sub

```

Sample VBA code for controlling Service Codes interface:

```

25 Code
1 VERSION 1.0 CLASS
2 BEGIN
3 MultiUse = -1 'True
4 END
30 5 Attribute VB_Name = "Form_ServiceCodes"
6 Attribute VB_GlobalNameSpace = False
7 Attribute VB_Creatable = True
8 Attribute VB_PredeclaredId = True
9 Attribute VB_Exposed = False
35 10 Option Compare Database
11 Option Explicit
12
13 Private Sub Command16_Click()
14 DoCmd.OpenReport "ServiceCodes By Code", acViewPreview
15 End Sub
40 16
17 Private Sub Command17_Click()
18 DoCmd.OpenReport "ServiceCodes By Description", acViewPreview
19 End Sub
20
45 21 Private Sub Form_Current()
22 If Me.NewRecord Then
23 ServiceCode.SetFocus
24 End If
25 End Sub
50 26
27 Private Sub Form_Load()
28 If Not IsNull(Me.OpenArgs) Then
29 Me![ServiceCode] = Me.OpenArgs
30 End If
55 31 End Sub

```

Function #2 - History Compilation

History compilation entails the process of taking bank statements from previous months and entering them into a computer database in the Bank Analyzer. Data collected

by this process is used for making accurate and informative comparisons between clients, and is helpful in determining appropriate and fair service charges. A flowchart of the process is shown in Fig. 22, and field layouts for a Bank Service Charge History table is shown in Fig. 23.

5 SECTION II - Dynamic Compilation - Input of Monthly Bank Statement Data

The Bank Statement issued by the bank to the client encompasses all stores for that client. The Bank Analyzer program provides a data collection form where this information is housed. This information is the raw material used to generate the final product. The two main tables used in this section are the AnalysisHdr and AnalysisDtl files. See Fig. 24 and Fig. 25 for the field layouts of these tables.

Sample VBA Code for controlling interface for AnalysisHdr:

```

Code
1  VERSION 1.0 CLASS
2  BEGIN
3  MultiUse = -1 'True
4  END
5  Attribute VB_Name = "Form_AnalysisHdr"
6  Attribute VB_GlobalNameSpace = False
7  Attribute VB_Creatable = True
8  Attribute VB_PredeclaredId = True
9  Attribute VB_Exposed = False
10 Option Compare Database
11 Option Explicit
12
13 Private Sub BankCode_BeforeUpdate(Cancel As Integer)
14 If Not Me.NewRecord Then
15 If MsgBox("You are about to change the bank code." & vbCr & "This will change all
service codes to be under the new bank." & vbCr & "Is this what you want to do?",
vbOKCancel, "Key Field Change") = vbCancel Then
16 DoCmd.CancelEvent
17 Me!BankCode.Undo
18 End If
19 End If
20
21 End Sub
22
23 Private Sub BankCode_GotFocus()
24 BankCode.DropDown
25 End Sub
26
27 Private Sub BankCode_NotInList(NewData As String, Response As Integer)
28 Dim Result
29 Dim msg As String
30
31 If NewData = "" Then Exit Sub 'cleared combo box
32
33 msg = "" & NewData & " is not in the Banks file." & vbCr & vbCr & "Do you want to add it?"
34 ' Yes
35 If MsgBox(msg, vbQuestion + vbYesNo) = vbYes Then
36 DoCmd.OpenForm "Banks", , , , acFormAdd, acDialog, NewData
37 End If

```



```

103 Private Sub Command60_Click()
104 Dim crt As String
105 crt = "[CustomerNo] = " & Me!CustomerNo & ""
106 crt = crt & " AND [RevDate] = #" & Me!RevDate & "#"
5 107 DoCmd.OpenReport "Bank Balance Data By EC Rate", acViewPreview, crt
108
109 End Sub
110
111 Private Sub Command61_Click()
112 DoCmd.OpenForm "EC Analysis Select"
10 113 End Sub
114
115 Private Sub CustomerNo_BeforeUpdate(Cancel As Integer)
116 If Not Me.NewRecord Then
15 117 If MsgBox("You are about to change Customer number." & vbCrLf & "This will result in
these service codes to be under the new customer." & vbCrLf & "Is this what you want to
do?", vbOKCancel, "Key Field Change") = vbCancel Then
118 DoCmd.CancelEvent
119 Me!CustomerNo.Undo
20 120 End If
121 End If
122
123 End Sub
124
25 125 Private Sub CustomerNo_GotFocus()
126 CustomerNo.DropDown
127 End Sub
128
30 129 Private Sub CustomerNo_NotInList(NewData As String, Response As Integer)
130 Dim Result
131 Dim msg As String
132
133 If NewData = "" Then Exit Sub ' cleared combo box
134
35 135 msg = "" & NewData & " is not in the Customer file." & vbCrLf & vbCrLf & "Do you want to add
' Yes
136 ' Yes
137 If MsgBox(msg, vbQuestion + vbYesNo) = vbYes Then
138 DoCmd.OpenForm "Customer", , , acFormAdd, acDialog, NewData
40 139 End If
140
141 ' look for new record added
142 Result = DLookup("[CustomerNo]", "Customer", "[CustomerNo] = " & NewData & "")
143 If IsNull(Result) Then
45 144 'suppress error message
145 Response = acDataErrContinue
146 MsgBox "Please enter a new customer number."
147 Else
148 ' they added it
50 149 Response = acDataErrAdded
150 End If
151
152
153 End Sub
154
55 155 Private Sub Form_Activate()
156 If CurrentUser() = "Admin" Or CurrentUser = "CarmJr" Then
157 DoCmd.ShowToolbar "Menu Bar", acToolbarYes
158 Else
159 DoCmd.ShowToolbar "Menu Bar", acToolbarNo
60 160 End If
161 MaximizeRestoredForm Forms!analysisHdr
162 End Sub
163
65 164 Private Sub Form_Close()
165 DoCmd.OpenForm "Analysis Select"
166 End Sub

```

```

167
168 Private Sub Form_Current()
169 If Me.NewRecord Then
170     CustomerN.SetFocus
171 End If
172 End Sub
173
174 Private Sub NegCollectedRate_Exit(Cancel As Integer)
175 If Me.NewRecord Then
176     AnalysisDt.SetFocus
177 End If
178 End Sub

```

SECTION III – Report Generation

This section describes the black box effect of the invention and discloses the methods and algorithms used to create the desired results. It also lists sample source code of key aspects of the program.

Once the data in the previous two sections has been amassed, it is now ready to be processed. This section illustrates the core functionality of the present invention. A majority of the information revealed by the reports was previously unknown to the clients.

The following lists Key fields which are utilized throughout the Bank Analyzer software:

AvgDlyPos	- the average daily positive balance
FedPct	- the current percentage mandated for the Reserve requirement
AdjBal	- adjusted balance
AvgBal	- average balance
AvgFloat	- average float
ECRate	- earnings credit rate
ReserveReq	- reserve requirement amount
AvailableBalance	- available balance
AvgFloatPerStore	- average float per store
AnalysisNo	- unique indexed number
NegCollectedRate	- negative collected rate, compared with the ECRate

Report: Bank Balance Data (Fig. 2)

Objective: Determine actual Total service charges paid by clients

The TotalServiceCharges field is derived from the CalcServTotal function, which is fed the information from the analyses entered during the dynamic compilation phase. The source code follows, which describes how to derive the actual service charges paid - by bank and with totals (2a), for the client being examined.

```

Public Function CalcServTotal(R As Report)
    ' Determining actual service charges

```

` This function is called for every line displayed and/or printed in the Bank Balance Data report – Fig. 2 in the disclosure

```
Dim tmpAmount As Double
Dim msg, crt As String
On Error Resume Next
```

5

```
` select the bank we're examining
crt = "[CustomerNo] = " & Forms!analysisHdr.CustomerNo & ""
crt = crt & " AND [BankCode] = " & R.BankCode & ""
crt = crt & " AND [RevDate] = #" & Forms!analysisHdr.RevDate & "#"
```

10

```
` obtain fields from the AnalysisDtl table described in Dynamic Compilation section
tmpAmount = DSum("[Activity]*[UnitPrice]", "AnalysisDtlQ", crt)
```

```
` return the answers in variables
ServicesTotal = ServicesTotal + tmpAmount
CalcServTotal = tmpAmount
```

15

```
On Error GoTo 0
End Function
```

Report: Earnings Credit Analysis (Fig. 3)

- Objectives:
1. Basis points shaved
 2. Projected Fed Fund amount
 3. Projected T-Bill amount
 4. Projected Income Difference

20

The report generation process begins with a data entry screen, as shown in Fig. 26. The T-Bill, Fed Funds, Interest Rate and Prime are entered prior to generating the report. These are the default values that the program uses in determining the objectives. These values can be customized by the client to allow for more accurate reporting. Results are revealed in a simple color-coded scheme allowing clients to see “at a glance” exposure to points of concern. (3e)

25

Exemplary source code follows:

To display the Basis points shaved using color codes: (3e)

30

```
Public Function ColorCodeBanks
Dim tmpAmt, tmpAmount As Double
PointsShaved.Visible = True
` turn Overdraft balances to Red
Select Case AvgBal
Case Is >= 0
    AvgBal.ForeColor = vbBlack
Case Else
    AvgBal.ForeColor = vbRed
End Select
```

35

```

' color code how many points are shaved
' constants representing colors are stored in an 'include' file
Select Case PointsShaved
Case Is >= 200
5   ' RED --> Worst offenders
    BankCode.ForeColor = vbRed
    BankName.ForeColor = vbRed
    PointsShaved.ForeColor = vbRed
Case 100 To 199
10   BankCode.ForeColor = vbCyan
    BankName.ForeColor = vbCyan
    PointsShaved.ForeColor = vbCyan
Case 26 To 99
15   BankCode.ForeColor = vbMagenta
    BankName.ForeColor = vbMagenta
    PointsShaved.ForeColor = vbMagenta
Case 1 To 25
20   BankCode.ForeColor = vbDarkGreen
    BankName.ForeColor = vbDarkGreen
    PointsShaved.ForeColor = vbDarkGreen
Case Is <= 0
25   PointsShaved.Visible = False
    BankCode.ForeColor = vbBlue
    BankName.ForeColor = vbBlue
    PointsShaved.ForeColor = vbBlue
End Select
End Function

```

Objective algorithms:

The Fed Funds Income difference is determined by taking the inputted current Fed Funds percentage and multiplying it by the Available Balance, then subtracting the annualized Earnings Credit amount.

The T-Bill difference is determined by taking the inputted current Treasury Bill rate and multiplying by the Available Balance, then subtracting the annualized Earnings Credit Amount.

The Interest Yield difference is determined by taking the inputted current Interest rate and multiplying by the Available Balance, then subtracting the annualized Earnings Credit Amount.

Key Source Code functions and examples: (3c)

```

FedFundsIncomeDifference=[Forms]![EC Analysis Select]![FedFunds]*[AvailableBal]-([EarningsCredit]*12)
40  TBillDifference=[Forms]![EC Analysis Select]![Tbill]*[AvailableBal]-([EarningsCredit]*12)
    IncomeDifference=[Forms]![EC Analysis Select]![InterestRate]*[AvailableBal]-([EarningsCredit]*12)

```

Source Code display:

```

45  SELECT IIf([AvgDlyPos]=0,([FedPct]/100)*[AdjBal],([FedPct]/100)*[AvgDlyPos]) AS ReserveReq,
    AnalysisHdr.CustomerNo, AnalysisHdr.BankCode, AnalysisHdr.RevDate, AnalysisHdr.AvgBal,
    AnalysisHdr.AvgFloat, AnalysisHdr.FedPct, AnalysisHdr.ECRate, [AvgDlyNeg]+[AdjBal] AS AvgDlyPos,

```

```

AnalysisHdr.AvgDlyNeg, Banks.BankName, Customer.CustomerName, [AvgBal]-[AvgFloat] AS AdjBal,
Banks.City, Banks.State, [AvgDlyPos]-[ReserveReq] AS AvailableBalance, [AvailableBalance]*[ECRate]/1200
AS EarningsCredit, AnalysisHdr.NumberOfStores, [AvgFloat]/[NumberOfStores] AS AvgFloatPerStore,
AnalysisHdr.AnalysisNo, IIf([AvgDlyPos]=0,[AdjBal]-[FedRes],[AvgDlyPos]-[FedRes]) AS AvailableBal,
IIf([AvgDlyPos]=0,([FedPct]/100)*[AdjBal],([FedPct]/100)*[AvgDlyPos]) AS FedRes,
AnalysisHdr.NegCollectedRate
FROM Customer INNER JOIN (Banks INNER JOIN AnalysisHdr ON Banks.BankCode =
AnalysisHdr.BankCode) ON Customer.CustomerNo = AnalysisHdr.CustomerNo;

```

Public Function CalcTBillTotal(R As Report)

```

Dim tmpAmount As Double
If R!TBillIncomeDifference > 0 Then
    tmpAmount = R!TBillIncomeDifference
    TBillDifference = TBillDifference + tmpAmount
End If
If R!FedFundsIncomeDifference > 0 Then
    FedFundsDifference = R!FedFundsIncomeDifference + FedFundsDifference
End If
If R!InterestYieldIncomeDifference > 0 Then
    InterestYieldDifference = R!InterestYieldIncomeDifference + InterestYieldDifference
End If

CalcTBillTotal = tmpAmount
End Function

```

To derive sums and annualized results, the totals are added for each line (corresponding to one client Bank), and then multiplied by 12 for annualization. The report's objectives are supported by displaying the Average Investable income and the Annualized Projected income. (3d)

Report: Earnings Credit And Negative Collected Rates (Fig. 4)

Objective: Match the Earnings Credit on positive balances to the rate imposed on Negative Collected funds.

The report generation process begins with a data entry screen, as shown in Fig. 27.

Objective algorithm:

Display the Negative collected rate on the same report with the Earnings Credit rate.

Exemplary Source code:

Selects and displays the data shown on Fig. 4.

```

SELECT IIf([AvgDlyPos]=0,([FedPct]/100)*[AdjBal],([FedPct]/100)*[AvgDlyPos]) AS ReserveReq,
AnalysisHdr.CustomerNo, AnalysisHdr.BankCode, AnalysisHdr.RevDate, AnalysisHdr.AvgBal,
AnalysisHdr.AvgFloat, AnalysisHdr.FedPct, AnalysisHdr.ECRate, [AvgDlyNeg]+[AdjBal] AS AvgDlyPos,
AnalysisHdr.AvgDlyNeg, Banks.BankName, Customer.CustomerName, [AvgBal]-[AvgFloat] AS AdjBal,

```

Banks.City, Banks.State, [AvgDlyPos]-[ReserveReq] AS AvailableBalance, [AvailableBalance]*[ECRate]/1200
 AS EarningsCredit, AnalysisHdr.NumberOfStores, [AvgFloat]/[NumberOfStores] AS AvgFloatPerStore,
 AnalysisHdr.AnalysisNo, IIf([AvgDlyPos]=0,[AdjBal]-[FedRes],[AvgDlyPos]-[FedRes]) AS AvailableBal,
 IIf([AvgDlyPos]=0,([FedPct]/100)*[AdjBal],([FedPct]/100)*[AvgDlyPos]) AS FedRes,
 AnalysisHdr.NegCollectedRate
 FROM Customer INNER JOIN (Banks INNER JOIN AnalysisHdr ON Banks.BankCode =
 AnalysisHdr.BankCode) ON Customer.CustomerNo = AnalysisHdr.CustomerNo;

Report: Checks Deposits and Float Data (Fig. 5)

- Objectives:
1. Determine actual availability date of client's monies (5a)
 2. Show abnormal Out Of District activity (5b)
 3. Verify immediate availability of On Us activity (5c)

Exemplary Source Code for report output:

The INNER JOIN ensures proper bank and client information extracted.

```

SELECT IIf([AvgDlyPos]=0,([FedPct]/100)*[AdjBal],([FedPct]/100)*[AvgDlyPos]) AS ReserveReq,
AnalysisHdr.CustomerNo, AnalysisHdr.BankCode, AnalysisHdr.RevDate, AnalysisHdr.AvgBal,
AnalysisHdr.AvgFloat, AnalysisHdr.FedPct, AnalysisHdr.ECRate, [AvgDlyNeg]+[AdjBal] AS AvgDlyPos,
AnalysisHdr.AvgDlyNeg, Banks.BankName, Customer.CustomerName, [AvgBal]-[AvgFloat] AS AdjBal,
Banks.City, Banks.State, [AvgDlyPos]-[ReserveReq] AS AvailableBalance, [AvailableBalance]*[ECRate]/1200
AS EarningsCredit, AnalysisHdr.NumberOfStores, [AvgFloat]/[NumberOfStores] AS AvgFloatPerStore,
AnalysisHdr.AnalysisNo, IIf([AvgDlyPos]=0,[AdjBal]-[FedRes],[AvgDlyPos]-[FedRes]) AS AvailableBal,
IIf([AvgDlyPos]=0,([FedPct]/100)*[AdjBal],([FedPct]/100)*[AvgDlyPos]) AS FedRes,
AnalysisHdr.NegCollectedRate, Customer.PDL1, Customer.PDL2, Customer.PDL3, Customer.PDL4,
AnalysisHdr.ChecksOutDist, AnalysisHdr.ChecksInDist, AnalysisHdr.MixedDeposits,
AnalysisHdr.DepositTickets, AnalysisHdr.ChecksOnUs,
IIf([TotalChecksFloated]>0,[AvgFloat]/[TotalChecksFloated]*30,0) AS AvgCheckSize,
[ChecksOutDist]+[ChecksInDist]+[MixedDeposits] AS TotalChecksFloated, AnalysisHdr.DepCost,
AnalysisHdr.BAICost, AnalysisHdr.DepReconCost, AnalysisHdr.NightBagsCost
FROM Customer INNER JOIN (Banks INNER JOIN AnalysisHdr ON Banks.BankCode =
AnalysisHdr.BankCode) ON Customer.CustomerNo = AnalysisHdr.CustomerNo
ORDER BY AnalysisHdr.AvgFloat DESC;
  
```

Sample VBA Code: Checks Deposits and Float Data

```

Code
1  VERSION 1.0 CLASS
2  BEGIN
3    MultiUse = -1 'True
4  END
5  Attribute VB_Name = "Report_Checks Deposits and Float Data"
6  Attribute VB_GlobalNameSpace = False
7  Attribute VB_Creatable = True
8  Attribute VB_PredeclaredId = True
9  Attribute VB_Exposed = False
10 Option Compare Database
11 Option Explicit
12
13 Private Sub GroupF   ter1_Print(Cancel As Integer, PrintCount As Integer)
14 '
15 '
16 End Sub
17
  
```

```

18 Private Sub Detail_Print(Cancel As Integer, PrintCount As Integer)
19 ' this is done in the Activate Event now
20 Dim tDT, tMD, tCOU, tCID, tCOD As Double
21 'Call CalcCDFData(Report, tDT, tMD, tCOU, tCID, tCOD)
22 'DepositTickets = tDT
23 'MixedDeposits = tMD
24 'ChecksOnUs = tCOU
25 'ChecksInDist = tCID
26 'ChecksOutDist = tCOD
27 End Sub
28
29 Private Sub Report_Activate()
30 ' generate the activity data
31 Call CalcCDFData
32 End Sub
33
34 Private Sub Report_Deactivate()
35 Application.Echo True
36 End Sub
37
38 Private Sub ReportFooter_Print(Cancel As Integer, PrintCount As Integer)
39 Dim tDT, tMD, tCOU, tCID, tCOD As Double
40 Call GetCheckDepTotals(Report, tDT, tMD, tCOU, tCID, tCOD)
41 SumDepositTickets = tDT
42 SumMixedDeposits = tMD
43 SumChecksOnUs = tCOU
44 SumChecksInDist = tCID
45 SumChecksOutDist = tCOD
46 End Sub

```

Report: Breakdown By Service Charge Groups (Fig. 6)

- Objectives:
1. Categorize Depository Costs (6a)
 2. Categorize Check Costs (6b)
 3. Categorize Account Maintenance Costs (6c)

Sample VBA Code: Breakdown By Service Charge Groups

For the report output:

```

SELECT IIf([AvgDlyPos]=0,([FedPct]/100)*[AdjBal],([FedPct]/100)*[AvgDlyPos]) AS ReserveReq,
AnalysisHdr.CustomerNo, AnalysisHdr.BankCode, AnalysisHdr.RevDate, AnalysisHdr.AvgFloat,
AnalysisHdr.FedPct, AnalysisHdr.ECRate, [AvgDlyNeg]+[AdjBal] AS AvgDlyPos, AnalysisHdr.AvgDlyNeg, Banks.BankName,
Customer.CustomerName, [AvgBal]-[AvgFloat] AS AdjBal, Banks.City, Banks.State, [AvgDlyPos]-[ReserveReq] AS
AvailableBalance, [AvailableBalance]*[ECRate]/1200 AS EarningsCredit, AnalysisHdr.NumberOfStores,
[AvgFloat]/[NumberOfStores] AS AvgFloatPerStore, AnalysisHdr.AnalysisNo, IIf([AvgDlyPos]=0,[AdjBal]-
[FedRes],[AvgDlyPos]-[FedRes]) AS AvailableBal,
IIf([AvgDlyPos]=0,([FedPct]/100)*[AdjBal],([FedPct]/100)*[AvgDlyPos]) AS FedRes, AnalysisHdr.NegCollectedRate,
Customer.PDL1, Customer.PDL2, Customer.PDL3, Customer.PDL4, AnalysisHdr.ChecksOutDist, AnalysisHdr.ChecksInDist,
AnalysisHdr.MixedDeposits, AnalysisHdr.DepositTickets, AnalysisHdr.ChecksOnUs,
IIf([TotalChecksFloated]>0,[AvgFloat]/[TotalChecksFloated]*30,0) AS AvgCheckSize,
[ChecksOutDist]+[ChecksInDist]+[MixedDeposits] AS TotalChecksFloated, AnalysisHdr.DepCost, AnalysisHdr.BAICost,
AnalysisHdr.DepReconCost, AnalysisHdr.NightBagsCost
FROM Customer INNER JOIN (Banks INNER JOIN AnalysisHdr ON Banks.BankCode = AnalysisHdr.BankCode) ON
Customer.CustomerNo = AnalysisHdr.CustomerNo
ORDER BY AnalysisHdr.AvgFloat DESC;

```

For the calculations:

Code
1 VERSION 1.0 CLASS


```

2 BEGIN
3 MultiUse = -1 True
4 END
5 Attribute VB_Name = "Rep rt_Breakdown By Service Charge Groups"
6 Attribute VB_GlobalNameSpace = False
7 Attribute VB_Creatable = True
8 Attribute VB_PredeclaredId = True
9 Attribute VB_Exposed = False
10 Option Compare Database
11 Option Explicit
12
13 Private Sub Detail_Print(Cancel As Integer, PrintCount As Integer)
14 Dim crt As String
15 crt = "[CustomerNo] = " & Forms!analysisHdr.CustomerNo & ""
16 crt = crt & " AND [BankCode] = " & Report.BankCode & ""
17 crt = crt & " AND [RevDate] = #" & Forms!analysisHdr.RevDate & "#"
18
19 AnnualServiceCharges = DSum("[ExtCharge]", "AnalysisDtlQ", crt) * 12
20 ' Calculate the totals
21 DepositCost = CalcCostTotal(crt, 2) * 12
22 CheckDepCost = CalcCostTotal(crt, 1) * 12
23 BalInfoCost = CalcCostTotal(crt, 3) * 12
24 TotalCost = DepositCost + CheckDepCost + BalInfoCost
25 ' derive %'s
26 DepositCostPer = DepositCost / AnnualServiceCharges
27 CheckDepCostPer = CheckDepCost / AnnualServiceCharges
28 BalInfoCostPer = BalInfoCost / AnnualServiceCharges
29 TotalCostPer = TotalCost / AnnualServiceCharges
30
31 End Sub
32
33 Private Sub ReportFooter_Print(Cancel As Integer, PrintCount As Integer)
34 Dim crt As String
35 crt = "[CustomerNo] = " & Forms!analysisHdr.CustomerNo & ""
36 crt = crt & " AND [RevDate] = #" & Forms!analysisHdr.RevDate & "#"
37
38 SAnnualServiceCharges = DSum("[ExtCharge]", "AnalysisDtlQ", crt) * 12
39 ' Calculate the totals
40 SDepositCost = CalcCostTotal(crt, 2) * 12
41 SCheckDepCost = CalcCostTotal(crt, 1) * 12
42 SBalInfoCost = CalcCostTotal(crt, 3) * 12
43 STotalCost = SDepositCost + SCheckDepCost + SBalInfoCost
44 ' derive %'s
45 sDepositCostPer = SDepositCost / SAnnualServiceCharges
46 sCheckDepCostPer = SCheckDepCost / SAnnualServiceCharges
47 SBalInfoCostPer = SBalInfoCost / SAnnualServiceCharges
48 STotalCostPer = STotalCost / SAnnualServiceCharges
49
50 End Sub

```

Report: Breakdown By User Supplied Groups (Fig. 7)

Objective: Categorize any user defined bank cost

To achieve this special objective, an additional Category table, shown in Fig. 28, is added to the database. This table works in conjunction with the Service Codes table shown in Fig. 17. A Category Select Form is also used in this process. A screen display of the form is shown in Fig. 29. The example in Fig. 29 shows categories, 60 (night bags), 62 (rolled coins) and 64 (return checks) as being selected.

Sample VBA Code: Breakdown By Service Charges Variable

C de

```

1  VERSION 1.0 CLASS
2  BEGIN
5  3  MultiUse = -1 'True
4  END
5  Attribute VB_Name = "Report_Breakdown By Service Charges Variable"
6  Attribute VB_GlobalNameSpace = False
7  Attribute VB_Creatable = True
10 8  Attribute VB_PredeclaredId = True
9  Attribute VB_Exposed = False
11 Option Compare Database
12 Option Explicit
13 Private Sub Detail_Print(Cancel As Integer, PrintCount As Integer)
14 Dim crt, CatCode1, CatCode2, CatCode3 As String
15 crt = "[CustomerNo] = '" & Forms!analysisHdr.CustomerNo & "'"
16 crt = crt & " AND [BankCode] = '" & Report.BankCode & "'"
17 crt = crt & " AND [RevDate] = #" & Forms!analysisHdr.RevDate & "#"
20 18
19 CatCode1 = Forms!CatSelect!CatCode1
20 CatCode2 = Forms!CatSelect!CatCode2
21 CatCode3 = Forms!CatSelect!CatCode3
22 AnnualServiceCharges = DSum("[ExtCharge]", "AnalysisDtlQ", crt) * 12
25 23 ' Calculate the totals
24 ' Var names are carried over from base report, they're actually Cat1, Cat2, Cat3
25 DepositCost = CalcCostTotal(crt, CatCode1) * 12
26 CheckDepCost = CalcCostTotal(crt, CatCode2) * 12
27 BalInfoCost = CalcCostTotal(crt, CatCode3) * 12
30 28 TotalCost = DepositCost + CheckDepCost + BalInfoCost
29 ' derive %'s
30 DepositCostPer = DepositCost / AnnualServiceCharges
31 CheckDepCostPer = CheckDepCost / AnnualServiceCharges
32 BalInfoCostPer = BalInfoCost / AnnualServiceCharges
35 33 TotalCostPER = TotalCost / AnnualServiceCharges
34
35 End Sub
36
40 37 Private Sub PageHeader_Print(Cancel As Integer, PrintCount As Integer)
38 Dim crt As String
39
40 crt = "[CategoryCode] = '" & Forms!CatSelect!CatCode1
41 CatLabel1 = DLookup("[Description]", "Category", crt)
42
45 43 crt = "[CategoryCode] = '" & Forms!CatSelect!CatCode2
44 CatLabel2 = DLookup("[Description]", "Category", crt)
45
46 crt = "[CategoryCode] = '" & Forms!CatSelect!CatCode3
47 CatLabel3 = DLookup("[Description]", "Category", crt)
50 48
49 End Sub
50
51 Private Sub Report_Activate()
52 DoCmd.Maximize
55 53 End Sub
54
55 Private Sub ReportFooter_Print(Cancel As Integer, PrintCount As Integer)
56 Dim crt, CatCode1, CatCode2, CatCode3 As String
57 crt = "[CustomerNo] = '" & Forms!analysisHdr.CustomerNo & "'"
60 58 crt = crt & " AND [RevDate] = #" & Forms!analysisHdr.RevDate & "#"
59
60 CatCode1 = Forms!CatSelect!CatCode1
61 CatCode2 = Forms!CatSelect!CatCode2
62 CatCode3 = Forms!CatSelect!CatCode3
65 63 SAnnualServiceCharges = DSum("[ExtCharge]", "AnalysisDtlQ", crt) * 12

```

```

64 ' Calculate the totals
65 SDep sitCost = CalcCostT tal(crt, CatCode2) * 12
66 SCheckDepCost = CalcCostTotal(crt, CatCode1) * 12
67 SBalInf C st = CalcCostT tal(crt, CatCode3) * 12
5 68 ST talC st = SDep sitC st + SCheckDepC st + SBalInfoC st
69 ' derive %'s
70 SDepositCostPer = SDepositCost / SAnnualServiceCharges
71 sCheckDepCostPer = SCheckDepCost / SAnnualServiceCharges
72 SBalInfoCostPer = SBalInfoCost / SAnnualServiceCharges
10 73 STotalCostPer = STotalCost / SAnnualServiceCharges
74 End Sub

```

This report uses the same query as Breakdown by Service Charge groups for Report selection. However, in this report, the users can supply their own categories.

Report: Unit Price Summary (Fig. 8)

15 Objective: Show every service charge assessed to a client

Sample code for the report output:

```

20 SELECT AnalysisHdr.CustomerNo, AnalysisHdr.BankCode, AnalysisHdr.RevDate, AnalysisDtl.ServiceCode,
    Banks.BankName, Banks.City, Banks.State, Customer.CustomerName, AnalysisDtl.Activity,
    AnalysisDtl.UnitPrice, ServiceCodes.ServiceDescription, [UnitPrice]*[Activity]/[Per] AS ExtAmt,
    ServiceCodes.Per, AnalysisDtl.ConCharge, AnalysisDtl.ConDate
FROM ServiceCodes INNER JOIN (((AnalysisDtl INNER JOIN Banks ON AnalysisDtl.BankCode =
    Banks.BankCode) INNER JOIN Customer ON AnalysisDtl.CustomerNo = Customer.CustomerNo) INNER JOIN
    AnalysisHdr ON (AnalysisHdr.RevDate = AnalysisDtl.RevDate) AND (AnalysisHdr.BankCode =
    AnalysisDtl.BankCode) AND (AnalysisHdr.CustomerNo = AnalysisDtl.CustomerNo) AND
25 (Customer.CustomerNo = AnalysisHdr.CustomerNo) AND (Banks.BankCode = AnalysisHdr.BankCode)) ON
    ServiceCodes.ServiceCode = AnalysisDtl.ServiceCode
WHERE (((AnalysisDtl.ServiceCode)<"95000"));

```

This report features a customizable form to further allow varying degrees of results. A sample screen display of the form is shown in Fig. 30. The following source code controls the flexible data entry form shown in Fig. 30. It allows for the user to selectively choose any combination of clients, banks, and/or service charges. It also allows for entry of date ranges and dollar figures, and provides a method of highlighting particular information once the report is generated.

```

35 Code
1 VERSION 1.0 CLASS
2 BEGIN
3 MultiUse = -1 True
4 END
5 Attribute VB_Name = "Form_AnalysisRpt Select"
40 6 Attribute VB_GlobalNameSpace = False
7 Attribute VB_Creatable = True
8 Attribute VB_PredeclaredId = True
9 Attribute VB_Exp sed = False
10 Option Compare Database
45 11 Option Explicit
12
13 Private Sub BankCheck_Click()

```



```

78 Next itm
79 If Len(crtBank) = 0 Then
80     MsgB x "No Banks selected."
81     Exit Sub
82 End If
83 Else
84     ' give us every bank
85     crtBank = "[BankCode] > '0'"
86 End If
87
10 88 ' get the Services
89 ' *****
90 Set ctlService = Me![ServiceList]
91 If Not ServiceCheck Then ' If they didn't select All services, what did they select?
92     For Each itm In ctlService.ItemsSelected
93         ' Build it
94         If Len(crtService) = 0 Then
95             crtService = "[ServiceCode] = '" & ctlService.ItemData(itm) & "'"
96         Else
97             crtService = crtService & " OR [ServiceCode] = '" & ctlService.ItemData(itm) & "'"
98         End If
99     Next itm
100 If Len(crtService) = 0 Then
101     MsgBox "No service codes selected."
102     Exit Sub
103 End If
104 Else
105     ' Give us every service code
106     crtService = "[ServiceCode] > '0'"
107 End If
108
30 109 ' get the customer accounts
110 ' *****
111 Set ctlCustomer = Me![CustomerList]
112 If Not CustomerCheck Then
113     For Each itm In ctlCustomer.ItemsSelected
114         If Len(crtCustomer) = 0 Then
115             crtCustomer = "[CustomerNo] = '" & ctlCustomer.ItemData(itm) & "'"
116         Else
117             crtCustomer = crtCustomer & " OR [CustomerNo] = '" & ctlCustomer.ItemData(itm)
40 & "'"
118         End If
119     Next itm
120 If Len(crtCustomer) = 0 Then
121     MsgBox "No customers selected."
122     Exit Sub
123 End If
124 Else
125     ' give us every customer
126     crtCustomer = "[CustomerNo] > '0'"
127 End If
128
129 ' Put it all together for banks, services, clients
130 crtAll = "(" & crtBank & ") AND (" & crtService & ") AND (" & crtCustomer & ")"
131 ' tack on date and dollar ranges
132 crtAll = crtAll & " AND [RevDate] >= #" & Me!StartDate & "# AND [RevDate] <= #" &
Me!EndDate & "#
133 crtAll = crtAll & " AND [UnitPrice] >= " & Val(Me!StartDollar) & " AND [UnitPrice] <= " &
60 Val(Me!EndDollar)
134
135 ' Run the rep rt with built criteria
136 If Sh wSummaryInfo Then
137     D Cmd.OpenRep rt "Analysis By Unit Price Summary", acViewPreview, , crtAll
138 Else
65 139     DoCmd.OpenRep rt "Analysis By Unit Price", acViewPreview, , crtAll
140 End If

```

```

141 End Sub
142
143 Private Sub Command17_Click()
144     D Cmd.Cl se acF rm, "AnalysisRpt Select"
5   145 End Sub
146
147 Private Sub CustomerCheck_Click()
148     ' if the user wants every Customer, don't let them select from the list box
10  149 Dim ctl As Control, itm As Variant
150 Set ctl = Me!CustomerList
151
152 If CustomerCheck Then
153     ' turn off anything they had selected
154     For Each itm In ctl.ItemsSelected
15
155         ctl.Selected(itm) = False
156     Next itm
157     ' turn the Customer List box and counter OFF
158     CustomerList.Enabled = False
159     CustomersSelected.Visible = False
20  160     ' turn the OK Button ON
161     Command16.Enabled = True
162 Else
163     ' turn the Customer List box and counter ON
164     CustomerList.Enabled = True
25  165     CustomersSelected.Visible = True
166 End If
167 End Sub
168
169 Private Sub CustomerList_Click()
30  170 BankList_Click
171 End Sub
172
173 Private Sub ServiceCheck_Click()
174 Dim ctl As Control, itm As Variant
35  175
176 ' if the user wants every Service, don't let them select from the list box
177 Set ctl = Me!ServiceList
178 If ServiceCheck Then
179     ' turn off anything they had selected
40  180     For Each itm In ctl.ItemsSelected
181         ctl.Selected(itm) = False
182     Next itm
183     ' turn off the whole List box and counter
184     ServiceList.Enabled = False
45  185     ServicesSelected.Visible = False
186
187     ' put the OK button back ON
188     Command16.Enabled = True
189 Else
50  190     ' turn the list box and counter back ON
191     ServiceList.Enabled = True
192     ServicesSelected.Visible = True
193 End If
194 End Sub
55  195
196 Private Sub ServiceList_Click()
197 BankList_Click
198 End Sub

```

Once the criteria are set, the report is generated.

60 The following sample code generates the actual report of Unit Price Summary (Fig. 8) and also controls the highlighting of particular information as specified by the user above.

Code

```
1  VERSION 1.0 CLASS
2  BEGIN
3  MultiUse = -1 'True
4  END
5  Attribute VB_Name = "Rep rt_Analysis By Unit Price"
6  Attribute VB_GlobalNameSpace = False
7  Attribute VB_Creatable = True
8  Attribute VB_PredeclaredId = True
9  Attribute VB_Exposed = False
10 Option Compare Database
11 Option Explicit
12
13 Private Sub Detail_Print(Cancel As Integer, PrintCount As Integer)
14 If Not IsNull(Forms![AnalysisRpt Select]!highlightCustNo) Then
15     If CustomerNo = Nz(Forms![AnalysisRpt Select]!highlightCustNo) Then
16         CustomerNo.ForeColor = 255
17         CustomerName.ForeColor = 255
18         BankName.ForeColor = 255
19         BankCode.ForeColor = 255
20         City.ForeColor = 255
21         State.ForeColor = 255
22         Activity.ForeColor = 255
23         UnitPrice.ForeColor = 255
24         ExtAmt.ForeColor = 255
25         RevDate.ForeColor = 255
26     Else
27         CustomerNo.ForeColor = 0
28         CustomerName.ForeColor = 0
29         BankName.ForeColor = 0
30         BankCode.ForeColor = 0
31         City.ForeColor = 0
32         State.ForeColor = 0
33         Activity.ForeColor = 0
34         UnitPrice.ForeColor = 0
35         ExtAmt.ForeColor = 0
36         RevDate.ForeColor = 0
37     End If
38 End If
39 End Sub
40
41 Private Sub Report_Activate()
42 DoCmd.Maximize
43 If Forms![AnalysisRpt Select]!ShowNames Then
44     CustomerName.Visible = True
45 Else
46     CustomerName.Visible = False
47 End If
48
49 End Sub
50
51 Private Sub ReportHeader_Print(Cancel As Integer, PrintCount As Integer)
52 Dim hCust As String
53 hCust = Nz(Forms![AnalysisRpt Select]!highlightCustNo, "")
54 If Len(hCust) > 0 Then
55     HighlightedCustomer = Forms![AnalysisRpt Select]!highlightCustNo
56     HighlightedCustomerName = DLookup("[CustomerName]", "Customer", "[CustomerNo]
= " & hCust & """)
57 Else
58     Label36.Visible = False
59 End If
60 End Sub
```

Report: Deposit Ticket Costs (Fig. 9)

C_de

```

1  VERSION 1.0 CLASS
2  BEGIN
5  3   MultiUse = -1 'True
4  END
5  Attribute VB_Name = "Report_Deposit Ticket Costs"
6  Attribute VB_GlobalNameSpace = False
10 7  Attribute VB_Creatable = True
8  Attribute VB_PredeclaredId = True
9  Attribute VB_Exposed = False
10 Option Compare Database
11 Option Explicit
12
15 13 Private Sub GroupFooter1_Print(Cancel As Integer, PrintCount As Integer)
14 '
15 '
16 End Sub
17
20 18 Private Sub Detail_Print(Cancel As Integer, PrintCount As Integer)
19 'setup our temp variables
20 Dim tD, tDepC, tBAI, tBAIO, tBAIC, tDR, tDRC, tDRO, tDepR, tDepRC, tNB, tNBC As Double
21 'go get totals
22 Call CalcDepTicketData(Report, tBAI, tBAIO, tDR, tDRO, tDepR, tDepRC, tNB)
25 23 'assign fields on report
24 'Deposit = tD
25 'DepCost = tDepC *** Commented fields are calculated in Activate proc
26 BAI = tBAI
27 'BAICost = tBAIC
30 28 BAIOther = tBAIO
29 DepRecon = tDR
30 'DepReconCost = tDRC
31 DepReconOther = tDRO
32 DepReorder = tDepR
35 33 DepReorderCost = tDepRC
34 NightBags = tNB
35 'NightBagsCost = tNBC
36 On Error Resume Next
37 If Loss1 > 0 Then Loss1.ForeColor = vbRed Else Loss1.ForeColor = vbBlack
40 38 If Loss2 > 0 Then Loss2.ForeColor = vbRed Else Loss2.ForeColor = vbBlack
39 If Loss3 > 0 Then Loss3.ForeColor = vbRed Else Loss3.ForeColor = vbBlack
40 If Loss4 > 0 Then Loss4.ForeColor = vbRed Else Loss4.ForeColor = vbBlack
41 On Error GoTo 0
42 End Sub
45 43
44 Private Sub Report_Activate()
45 Call WriteDepTicketData
46 End Sub
47
50 48 Private Sub Report_Deactivate()
49 Application.Echo True
50 End Sub

51
55 52 Private Sub ReportFooter1_Print(Cancel As Integer, PrintCount As Integer)
53 Dim t1, t2, t3, t4, t5, t6, t7, t8, t9, t10, t11, t12 As Double
54 Call GetDepTicketTotals(Report, t1, t2, t3, t4, t5, t6, t7, t8, t9, t10, t11, t12)
55 SumDeposit = t1
56 SumDepCost = t2
57 SumBAI = t3
60 58 SumBAIC st = t4
59 SumBAIOther = t5
60 SumDepRecon = t6
61 SumDepReconCost = t7

```



```

62 SumDepReconOther = t8
63 SumDepReorder = t9
64 SumDepReorderCost = t10
65 SumNightBags = t11
5 66 SumNightBagsCost = t12
67
68
69 End Sub
70
10 71 Private Sub ReportHeader_Print(Cancel As Integer, PrintCount As Integer)
72 InitVars
73 End Sub

```

Multiplying the SumDeposit field in line 55 above by 12 provides the projected Annual cost incurred for one year's worth of deposit costs. (9a)

15 The BreakEvenDeposit (9a) is determined by the following formula:

$$\frac{(((\text{DepCost}) + [\text{BAICost}] + [\text{DepReconCost}] + [\text{NightBagsCost}]) / [\text{DepositTickets}]) * 365 / [\text{Forms}]![\text{DepCost Select}]![\text{ReturnRate}]}$$

where the ReturnRate is a value inputted by the user (9e) which represents a typical net profit percentage for the particular client.

20 The values referred to in (9c) are stored in the client's profile in the database.

Lines 37 through 40 of the code determine the appropriate color for values referenced in (9d).

Report: Cash Activity Costs (Fig. 10)

Sample VBA Code for CashActivity Costs

```

25 Private Sub Detail_Print(Cancel As Integer, PrintCount As Integer)
Dim crt As String
crt = "[CustomerNo] = " & Forms!analysisHdr.CustomerNo & """"
crt = crt & " AND [BankCode] = " & Report.BankCode & """"
crt = crt & " AND [RevDate] = #" & Forms!analysisHdr.RevDate & "#"

30 DepPerUnitCost = 0 ' reset it- so it doesn't dupe
' Cash Deposit Per 1000
CashDepPer1000 = CalcDepTotal(crt, 101)
CashCostPer1000 = CalcCostTotal(crt, 101)
If CashDepPer1000 <> 0 Then
35 ' avg unit cost has to multiply back by Per1000
DepPerUnitCost = (CashCostPer1000 / CashDepPer1000) * 1000    Fig. 10, item 10b
End If
' Cash Strapped
DepStrappedUnitCost = 0
40 CashDepStrapped = CalcDepTotal(crt, 102)
DepStrappedCost = CalcCostTotal(crt, 102)
If DepStrappedCost <> 0 Then
DepStrappedUnitCost = DepStrappedCost / CashDepStrapped
End If

```

```

' Cash Loose
DepLooseUnitCost = 0
CashDepLoose = CalcDepTotal(crt, 103)
DepLooseCost = CalcCostTotal(crt, 103)
5 If CashDepLoose <> 0 Then
    DepLooseUnitCost = DepLooseCost / CashDepLoose
End If
' Other Cash costs
OtherCashUnitCost = 0 ' so it doesn't dupe
10 OtherCash = CalcDepTotal(crt, 104)
OtherCashTotal = CalcCostTotal(crt, 104)
If OtherCash <> 0 Then
    OtherCashUnitCost = OtherCashTotal / OtherCash
End If
15 ' Last few columns
'NightBagsCost = CalcCostTotal(crt, 60)
AvgCashPerStore = CashDepPer1000 / NumberOfStores
CashOrderCost = CalcCostTotal(crt, 61)
20 RolledCoinCost = CalcCostTotal(crt, 62) (Fig. 10, item 10b)
BoxCoinCost = CalcCostTotal(crt, 63)

End Sub

```

Multiplying the results from (10b) * 12 yields the results in (10c).

Report: Banking Activity (Fig. 11)

Objective: Summarize information

25 This report summarizes information presented in other reports. It provides one place to view Deposit Ticket costs, BAI Detail costs, Dep Recon costs, Night Bag costs, and Rolled Coin Costs.

```

30 Code
1 VERSION 1.0 CLASS
2 BEGIN
3 MultiUse = -1 True
4 END
5 Attribute VB_Name = "Report_Banking Activity"
6 Attribute VB_GlobalNameSpace = False
35 7 Attribute VB_Creatable = True
8 Attribute VB_PredeclaredId = True
9 Attribute VB_Exposed = False
10 Option Compare Database
11 Option Explicit
40 12
13 Private Sub Detail_Print(Cancel As Integer, PrintCount As Integer)
14 Dim crt As String
15 crt = "[CustomerNo] = '" & Forms!analysisHdr.CustomerNo & "'"
16 crt = crt & " AND [BankCode] = '" & Report.BankCode & "'"
45 17 crt = crt & " AND [RevDate] = #" & F rms!analysisHdr.RevDate & "#"
18
19 Dep sitTickets = CalcDepT tal(crt, 20)
20 BAIDetail = CalcDepT tal(crt, 40)

```

```

21 D pRecon = CalcDepTotal(crt, 45)
22 DepErrors = CalcDepTotal(crt, 46)
23 ChecksDeposited = CalcDepTotal(crt, 1)
24 CashDeposited = CalcDepTotal(crt, 101)
5 25 NightBags = CalcDepTotal(crt, 60)
26 CashOrdered = CalcDepTotal(crt, 61)
27 RolledCoin = CalcDepTotal(crt, 62)
28 BoxCoin = CalcDepTotal(crt, 63)
10 29 ReturnChecks = CalcDepTotal(crt, 64)
30 ReturnChecksRedep = CalcDepTotal(crt, 65)
31 ReturnCheckBuyBack = CalcDepTotal(crt, 66)
32 NSFUNCOD = CalcDepTotal(crt, 67)
33
34
15 35 End Sub
36
37 Private Sub Report_Activate()
38 DoCmd.Maximize
39 End Sub
20 40
41 Private Sub ReportFooter_Print(Cancel As Integer, PrintCount As Integer)
42 ' Carmen J. DeLeo July 8, 1999
43 Dim crt As String
44 ' No bank criteria on Sums
25 45 crt = "[CustomerNo] = '" & Forms!analysisHdr.CustomerNo & "'"
46 crt = crt & " AND [RevDate] = #" & Forms!analysisHdr.RevDate & "#"
47 SDepositedTickets = CalcDepTotal(crt, 20)
48 SBAIDetail = CalcDepTotal(crt, 40)
30 49 SDepRecon = CalcDepTotal(crt, 45)
50 SDepErrors = CalcDepTotal(crt, 46)

51 SChecksDeposited = CalcDepTotal(crt, 1)
52 SCashDeposited = CalcDepTotal(crt, 101)
53 SNightBags = CalcDepTotal(crt, 60)
35 54 SCashOrdered = CalcDepTotal(crt, 61)
55 SRolledCoin = CalcDepTotal(crt, 62)
56 SBoxCoin = CalcDepTotal(crt, 63)
57 SReturnChecks = CalcDepTotal(crt, 64)
58 SReturnChecksRedep = CalcDepTotal(crt, 65)
40 59 SReturnCheckBuyback = CalcDepTotal(crt, 66)
60 SNSFUNCOD = CalcDepTotal(crt, 67)
61
62 End Sub

```

CalcDepTotal function is used in many of the reports:

```

45 Public Function CalcDepTotal(crt As String, CatCode)
Dim tmpAmount, tmpCost As Double
Dim msg, crtAll As String

crtAll = crt & " AND [CategoryCode] = " & CatCode
tmpAmount = Nz(DSum("[Activity]", "AnalysisDtlSubTotalsQ", crtAll), 0)
CalcDepTotal = tmpAmount

50 End Function

```

This function is vital to determining the totals stored in the underlying Analysis Detail tables. As illustrated by the source code, this routine processes a category, then returns the total based on the Service and Category tables.

Report: Analysis By Unit Price (Fig. 12)

The source code is equivalent to the above report, except for the output.

Note: The Detail section of this report is not hidden, thereby showing the supporting data.

```
SELECT AnalysisHdr.CustomerNo, AnalysisHdr.BankCode, AnalysisHdr.RevDate, AnalysisDtl.ServiceCode,  
Banks.BankName, Banks.City, Banks.State, Customer.CustomerName, AnalysisDtl.Activity,  
AnalysisDtl.UnitPrice, ServiceCodes.ServiceDescription, [UnitPrice]*[Activity]/[Per] AS ExtAmt,  
ServiceCodes.Per, AnalysisDtl.ConCharge, AnalysisDtl.ConDate  
FROM ServiceCodes INNER JOIN (((AnalysisDtl INNER JOIN Banks ON AnalysisDtl.BankCode =  
Banks.BankCode) INNER JOIN Customer ON AnalysisDtl.CustomerNo = Customer.CustomerNo) INNER JOIN  
AnalysisHdr ON (AnalysisHdr.RevDate = AnalysisDtl.RevDate) AND (AnalysisHdr.BankCode =  
AnalysisDtl.BankCode) AND (AnalysisHdr.CustomerNo = AnalysisDtl.CustomerNo) AND  
(Customer.CustomerNo = AnalysisHdr.CustomerNo) AND (Banks.BankCode = AnalysisHdr.BankCode)) ON  
ServiceCodes.ServiceCode = AnalysisDtl.ServiceCode  
WHERE (((AnalysisDtl.ServiceCode)<"95000"));
```

DETERMINING PARTICULAR CATEGORY COSTS

Fig. 31 and Fig. 32 show excerpts from charts that disclose the method for determining a particular category's cost, using Night Bags as an example. This functionality is used frequently throughout the software.

The following code will determine the total cost. It assumes that the statement data has been keyed in, during the Dynamic Compilation phase.

```
Public Function CalcCostTotal(crt, CatCode)  
' return the cost of a group of service charges  
Dim tmpAmount, tmpCost As Double  
Dim msg, crtAll As String  
  
crtAll = crt & " AND [CategoryCode] = " & CatCode  
tmpAmount = Nz(DSum("[ExtCharge]", "AnalysisDtlSubTotalsQ", crtAll), 0)  
CalcCostTotal = tmpAmount  
End Function
```

This algorithm applies directly to Breakdown By Service Charge, and by User Supplied Groups.

SECTION IV – ILLUMINATION (Fig. 13 and Fig. 14)

Fig. 33 is a screen display of a Working Sheet Form that is used to summarize ServiceDescription items, and their actual costs and potential cost savings.

The Bank Analyzer described above allows a client to obtain information that they have never been able to access. In doing so, it permits the client to eliminate wasteful banking transactions, reduce the amount of transactions and obtain a reduction in bank service charges and other charges and fees.

The Bank Analyzer is preferably used by a third party entity, who has a plurality of clients who provide the third party entity with their respective bank information for analysis. The third party entity may then build up a historical database of bank charges and credits from a plurality of different banks, and can then provide accurate comparisons of a specific client's bank analysis. However, the Bank Analyzer may also be sold or licensed by the third party entity to a client, along with a database of current bank data so that the client may perform their own analyses.

The present invention may be implemented with any combination of hardware and software. The present invention can be included in an article of manufacture (e.g., one or more computer program products) having, for instance, computer useable media. The media has embodied therein, for instance, computer readable program code means for providing and facilitating the mechanisms of the present invention. The article of manufacture can be included as part of a computer system or sold separately.

It will be appreciated by those skilled in the art that changes could be made to the embodiments described above without departing from the broad inventive concept thereof. It is understood, therefore, that this invention is not limited to the particular embodiments disclosed, but it is intended to cover modifications within the spirit and scope of the present invention.

What is claimed is: